Stoma_Project

Valentina Stoma

11/26/2019

```
library(rvest)
```

```
## Loading required package: xml2
```

```
library(stringr)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- tidy
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v forcats 0.4.0
## v readr   1.3.1
```

```
## -- Conflicts ------------------------------------------------------------------------- tidyverse_
## x dplyr::filter()         masks stats::filter()
## x readr::guess_encoding() masks rvest::guess_encoding()
## x dplyr::lag()            masks stats::lag()
## x purrr::pluck()          masks rvest::pluck()
```

```
library(tidyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(RSQLite)
library(sqldf)
```

```
## Loading required package: gsubfn
```

```
## Loading required package: proto
```

```
## Warning in doTryCatch(return(expr), name, parentenv, handler): unable to load shared object '/Library
##   dlopen(/Library/Frameworks/R.framework/Resources/modules//R_X11.so, 6): Library not loaded: /opt/X1
##   Referenced from: /Library/Frameworks/R.framework/Resources/modules//R_X11.so
##   Reason: image not found
```

```
## Could not load tcltk.  Will use slower R code instead.
```

```
library(readxl)
library(ggpubr)
```

```
## Loading required package: magrittr
```

```
##
## Attaching package: 'magrittr'
```

```
## The following object is masked from 'package:purrr':
##
```

```
##     set_names
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```
library(usmap)
library(ggplot2)
```

```
#reading in the files
parks<-read_csv("Wiki_parks.csv")
```

```
## Parsed with column specification:
## cols(
##   url = col_character(),
##   Name = col_character(),
##   Name_link = col_character(),
##   State = col_character(),
##   Date_established = col_character(),
##   Area = col_character()
## )
```

```
#deleting unneccessary columns that are an accessaory of using Import.io
parks<- parks[,-c(1,3)]
head(parks)
```

```
## # A tibble: 6 x 4
##   Name           State          Date_established  Area
##   <chr>          <chr>          <chr>             <chr>
## 1 Acadia         Maine          February 26, 1919 49,075.26 acres (198.6 km
## 2 American Samoa American Samoa October 31, 1988  8,256.67 acres (33.4 km
## 3 Arches         Utah           November 12, 1971 76,678.98 acres (310.3 km
## 4 Badlands       South Dakota   November 10, 1978 242,755.94 acres (982.4 km
## 5 Big Bend       Texas          June 12, 1944     801,163.21 acres (3,242.2 km
## 6 Biscayne       Florida        June 28, 1980     172,971.11 acres (700.0 km
```

```
#creating a separate column to contain the copy of the  Area column for further modifications
# and changing the format of the Date_estblished column into Date
parks <- parks %>%
  mutate(Area_km = Area) %>%
  mutate(Date_established = mdy(Date_established))

head(parks)
```

```
## # A tibble: 6 x 5
##   Name        State      Date_established Area               Area_km
##   <chr>       <chr>      <date>           <chr>              <chr>
## 1 Acadia      Maine      1919-02-26       49,075.26 acres (~ 49,075.26 acres (1~
## 2 American S~ American ~ 1988-10-31       8,256.67 acres (3~ 8,256.67 acres (33~
## 3 Arches      Utah       1971-11-12       76,678.98 acres (~ 76,678.98 acres (3~
## 4 Badlands    South Dak~ 1978-11-10       242,755.94 acres ~ 242,755.94 acres (~
## 5 Big Bend    Texas      1944-06-12       801,163.21 acres ~ 801,163.21 acres (~
## 6 Biscayne    Florida    1980-06-28       172,971.11 acres ~ 172,971.11 acres (~
```

```
#cleaning up the area column in order to obtain only the acres in one column
clean_parks<- parks %>%
  mutate(Area = str_replace(Area, "acres \\(", " ")) %>%
  mutate(Area = str_replace(Area, "km", "")) %>%
```

```
  separate(., Area, into=c("Area", "Area_new"), sep=" ")
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 61 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
#getting rid of unnecessary columns
clean_parks<-clean_parks[,-c(5,6)]
#My dplyr package is conflicting with some other packcages so for some of the
#functions I need to specify to which package it belongs.
clean_parks<- clean_parks %>%
  dplyr::rename(Area_acres = Area)
```

Issue arose with date format for the introduction into the SQL - needed to change it into charachter in the following manner in order to insert into SQL

```
#this format change allows me to have normal character format that will
#be read correctly in the SQL database
clean_parks$Date_established<- as.character(as.Date(as.character(clean_parks$Date_established),
                                                    format = "%Y-%m-%d"))

#Using this str_trim, I trim out any lagging and leading spaces, which prohibit recognition
#of the same names in different table in the database
#additionally, deleting any spaces in the acres size for proper formatting.
clean_parks <- clean_parks %>%
  mutate_if(is.character, str_trim) %>%
  mutate(Area_acres = str_replace(Area_acres, ",", ""))
#moreover, I am rounding all the acres sizing of the national parks for universal fomatting.
clean_parks<-clean_parks %>%
  mutate(Area_acres = str_replace_all(Area_acres, "[.*+]", ""))
```

10 MOST VISITED PARKS IN THE US

```
#reading in the website
visitors<- read_html("https://www.nps.gov/aboutus/visitation-numbers.htm")

#extracting names, visitors from the read in website
names <-visitors %>%
  html_nodes(".CS_Element_Custom~ .CS_Element_Custom+ .CS_Element_Custom thead+ tbody td:nth-child(2)") %>%
  html_text

number <- visitors %>%
  html_nodes("#cs_control_5546313 td~ td+ td") %>%
  html_text()


#combining the extracted information into a dataframe.
most_visits<- data.frame(names, number)
#clean up the creatde dataframe to have appropriate formats for number and characters,
#as well as formatting the numbers to not have any commas. For the universal notation in this project,
#the names of the National park do not contain any other identifications except its actual name.
most_visits<-most_visits %>%
  mutate(number = as.character(number)) %>%
  mutate(number = str_replace_all(number, ",", "")) %>%
  mutate(number = as.numeric(number)) %>%
```

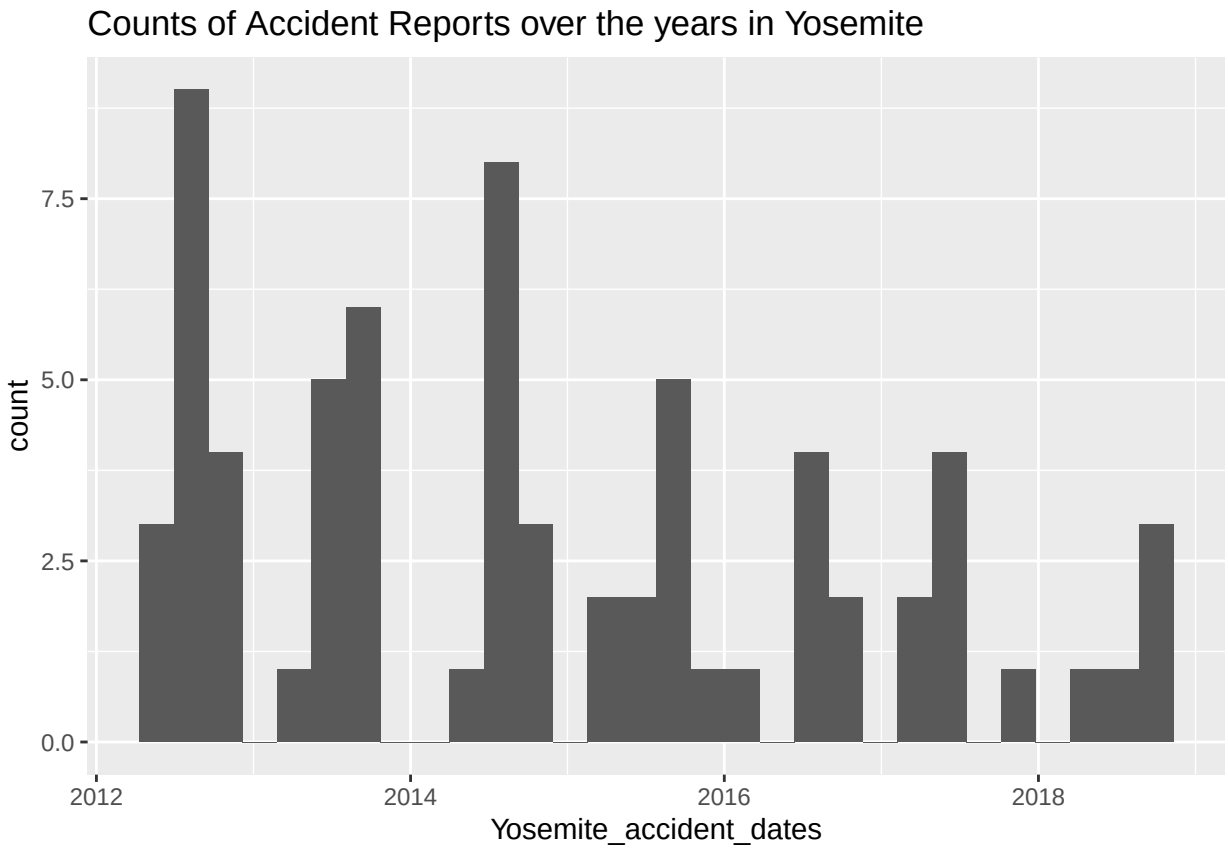
```
  mutate(names = str_replace(names, "National Park", "")) %>%
  dplyr::rename(visitor_number = number) %>%
  dplyr::rename(Park_name = names)
#Deleting any lagging and leading spaces.
most_visits <- most_visits %>%
  mutate_if(is.character, str_trim)
head(most_visits)
```

```
##               Park_name visitor_number
## 1 Great Smoky Mountains       15223697
## 2          Grand Canyon       14690418
## 3        Rocky Mountain       11421200
## 4                  Zion        9243305
## 5           Yellowstone        7804683
## 6              Yosemite        7578958
```

LIST OF YOSEMITE ACCIDENTS OVER THE YEARS

```
#reading in the web page with the accidents.
yosemite_accident<-read_html("https://www.nps.gov/yose/blogs/psarblog.htm")

#extracting the necessary information from the appropriate nodes.
Yosemite_accident_dates<-yosemite_accident %>%
  html_nodes(".date") %>%
  html_text()

description<-yosemite_accident %>%
  html_nodes(".slug p") %>%
  html_text()
#creating a list of just the names of the park these accidents refer to.
yose_name<-replicate(69, "Yosemite")
#this dataframe will not be included in the database, this is an example
#of missing data from other National Parks, and should be supplemented later on.
yose_acc<- data.frame(yose_name, Yosemite_accident_dates, description)
#modyifying the format of the date column as Date.
yose_acc<-yose_acc %>%
  mutate(Yosemite_accident_dates = mdy(Yosemite_accident_dates)) %>%
  dplyr::rename(accident_description = description)

#saving the dataframe to an appropriate name and modying the date format to be read into the
#SQL database in case in the future it will be supplemented with more national park accident data.
yosemite_accidents<- yose_acc
yosemite_accidents$Yosemite_accident_dates<- as.character(as.Date(as.character(yosemite_accidents$Yosem
                                                                               format = "%Y-%m-%d"))

#this plot is generated to demostrate when the accidents occur and to show that
#the data can be used in the analysis if more is collected accessibly by the national parks.
ggplot(yose_acc, aes(Yosemite_accident_dates)) +
  geom_histogram() +
  ggtitle("Counts of Accident Reports over the years in Yosemite")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



YELLOWSTONE TRAIL NAME and DIFFICULTY INFORMATION

```
#reading the web site containing the trail information.
yellowstone_trail<-read_html("https://www.yellowstonenationalparklodges.com/connect/yellowstone-hot-spo

#findign and reading in the nodes that have the inforamtion about names and trail difficulty.
trail_names<-yellowstone_trail %>%
  html_nodes("h3 strong") %>%
  html_text()

yellowstone_diff<- yellowstone_trail %>%
  html_nodes("p:nth-child(18) , p:nth-child(23) , p:nth-child(34) , p:nth-child(39) , p:nth-child(44) ,
  html_text()


yellow_loc<- yellowstone_trail %>%
  html_nodes("p:nth-child(65) , p:nth-child(60) , p:nth-child(55) , p:nth-child(50) , p:nth-child(45) ,
  html_text()
#creating a list containig the name of the national park for which I obtained the trail
#information. 12 is the number of trails I found from one source.
yellow_name<-replicate(12, "Yellowstone")
yellow_trails<- data.frame(yellow_name,trail_names, yellowstone_diff, yellow_loc)

#changing the content of one of the columns because of the way that the nodes read into R
```

```
yellow_trails <- yellow_trails %>%
  mutate(yellowstone_diff = str_replace(yellowstone_diff, "Location: Canyon Area", "Level of Difficulty
#cleaning up the obtained dataframe to reduce text in the level of difficulty column
#and to standardize the classification of trail level. Additioanlly, changing the names
#of the columns to accomodate addition of other national park trail information.
yellowstone_trails <- yellow_trails %>%
  mutate(yellowstone_diff = str_replace(yellowstone_diff, "Level of [Dd]ifficulty:", "")) %>%
  mutate(yellowstone_diff = str_replace(yellowstone_diff, "Difficulty:", "")) %>%
  dplyr::rename(difficulty_level = yellowstone_diff) %>%
  mutate(trail_names = str_to_sentence(trail_names)) %>%
  mutate(yellow_loc = str_replace(yellow_loc, "Location: ", "")) %>%
  dplyr::rename(trail_location = yellow_loc) %>%
  dplyr::rename(park_name = yellow_name) %>%
  dplyr::rename(yellowstone_trail_name = trail_names)
```

Elevation of parks

```
#this dataframe was obtained from Import.io
#there is some missing values in the Mountain range column in the dataframe so just in case replcae it
eleva_w_parks<- read.csv("Elevation_wiki_parks-(Crawl-Run)---2019-11-30T145634Z.csv", na.strings=c("","
head(eleva_w_parks)
```

```
##            Park_name         Peak_name        Mountain_range
## 1             Denali            Denali          Alaska Range
## 2 Wrangell-St. Elias Mount Saint Elias Saint Elias Mountains
## 3        Glacier Bay Mount Fairweather Saint Elias Mountains
## 4            Sequoia     Mount Whitney         Sierra Nevada
## 5      Mount Rainier     Mount Rainier         Cascade Range
## 6     Rocky Mountain        Longs Peak           Front Range
##          Mim_elevation       Vertical.relief
## 1      240 feet (73 m) 20,070 feet (6,120 m)
## 2        0 feet (0 m) 18,008 feet (5,489 m)
## 3        0 feet (0 m) 15,300 feet (4,700 m)
## 4   1,360 feet (410 m) 13,145 feet (4,007 m)
## 5   1,610 feet (490 m) 12,801 feet (3,902 m)
## 6 7,630 feet (2,330 m)  6,629 feet (2,021 m)
```

```
#Formatting the type of columns in the dataframe.
elevation<-eleva_w_parks %>%
  mutate(Mim_elevation = as.character(Mim_elevation)) %>%
  mutate(Vertical.relief = as.character(Vertical.relief))
elevation= as.tibble(elevation)
```

```
## Warning: `as.tibble()` is deprecated, use `as_tibble()` (but mind the new semantics).
## This warning is displayed once per session.
```

```
#renaming the column because fof the typo from Import.io
elevation<-elevation %>%
  dplyr::rename(Min_elevation = Mim_elevation)

head(elevation)
```

```
## # A tibble: 6 x 5
##   Park_name     Peak_name     Mountain_range   Min_elevation   Vertical.relief
```

```
##   <fct>          <fct>          <fct>            <chr>           <chr>
## 1 Denali         Denali         Alaska Range     240 feet (73 m) 20,070 feet (6,1~
## 2 Wrangell-St.~ Mount Saint ~ Saint Elias Mou~ 0 feet (0 m)    18,008 feet (5,4~
## 3 Glacier Bay    Mount Fairwe~ Saint Elias Mou~ 0 feet (0 m)    15,300 feet (4,7~
## 4 Sequoia        Mount Whitney Sierra Nevada    1,360 feet (41~ 13,145 feet (4,0~
## 5 Mount Rainier Mount Rainier Cascade Range    1,610 feet (49~ 12,801 feet (3,9~
## 6 Rocky Mounta~ Longs Peak     Front Range      7,630 feet (2,~ 6,629 feet (2,02~
```

We can calculate the final elevation at the peak level from the provided min_elevation (ground level for the park) and the vertical.relief (vertical chanhe experienced by the park).

```
#cleaning up the data further to only contain one column with clean values of
#min_elevation and vertical.relief  in feet
elev<- elevation %>%
  mutate(Min_elevation = str_replace_all(Min_elevation, "feet", " " )) %>%
  mutate(Min_elevation = str_replace(Min_elevation, "m", "")) %>%
  mutate(Min_elevation = str_replace(Min_elevation, "\\(", "")) %>%
  mutate(Min_elevation = str_replace(Min_elevation, "\\)", "")) %>%
  separate(., Min_elevation, into = c("Min_elevation_feet", "Min_elevation_m"),
           sep = " ", remove = TRUE)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 56 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
elev<- elev[, -c(5)]

elev<-elev %>%
  mutate(Vertical.relief = str_replace_all(Vertical.relief, "feet", " " )) %>%
  mutate(Vertical.relief = str_replace(Vertical.relief, "m", "")) %>%
  mutate(Vertical.relief = str_replace(Vertical.relief, "\\(", "")) %>%
  mutate(Vertical.relief = str_replace(Vertical.relief, "\\)", "")) %>%
  separate(., Vertical.relief, into= c("Vertical_relief_feet", "Vertical.relief.m"),
           sep = " ", remove = TRUE)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 56 rows [1, 2, 3, 4,
## 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```
elevation_all_parks<- elev[, -c(6)]
head(elevation_all_parks)
```

```
## # A tibble: 6 x 5
##   Park_name    Peak_name    Mountain_range    Min_elevation_f~ Vertical_relief_~
##   <fct>        <fct>        <fct>             <chr>            <chr>
## 1 Denali       Denali       Alaska Range      240              20,070
## 2 Wrangell-St~ Mount Saint~ Saint Elias Moun~ 0                18,008
## 3 Glacier Bay  Mount Fairw~ Saint Elias Moun~ 0                15,300
## 4 Sequoia      Mount Whitn~ Sierra Nevada     1,360            13,145
## 5 Mount Raini~ Mount Raini~ Cascade Range     1,610            12,801
## 6 Rocky Mount~ Longs Peak   Front Range       7,630            6,629
```

Calculating the final elevation :

```
#finally, calcualting the final elevation level for the peaks from the presented data.
elev_all_parks<-elevation_all_parks %>%
  mutate(Min_elevation_feet = str_replace_all(Min_elevation_feet, ",", "")) %>%
  mutate(Vertical_relief_feet = str_replace_all(Vertical_relief_feet, ",", "")) %>%

  mutate(Min_elevation_feet = as.integer(Min_elevation_feet)) %>%
```

```
  mutate(Vertical_relief_feet = as.integer(Vertical_relief_feet)) %>%
  mutate(Peak_height = Min_elevation_feet + Vertical_relief_feet)
```

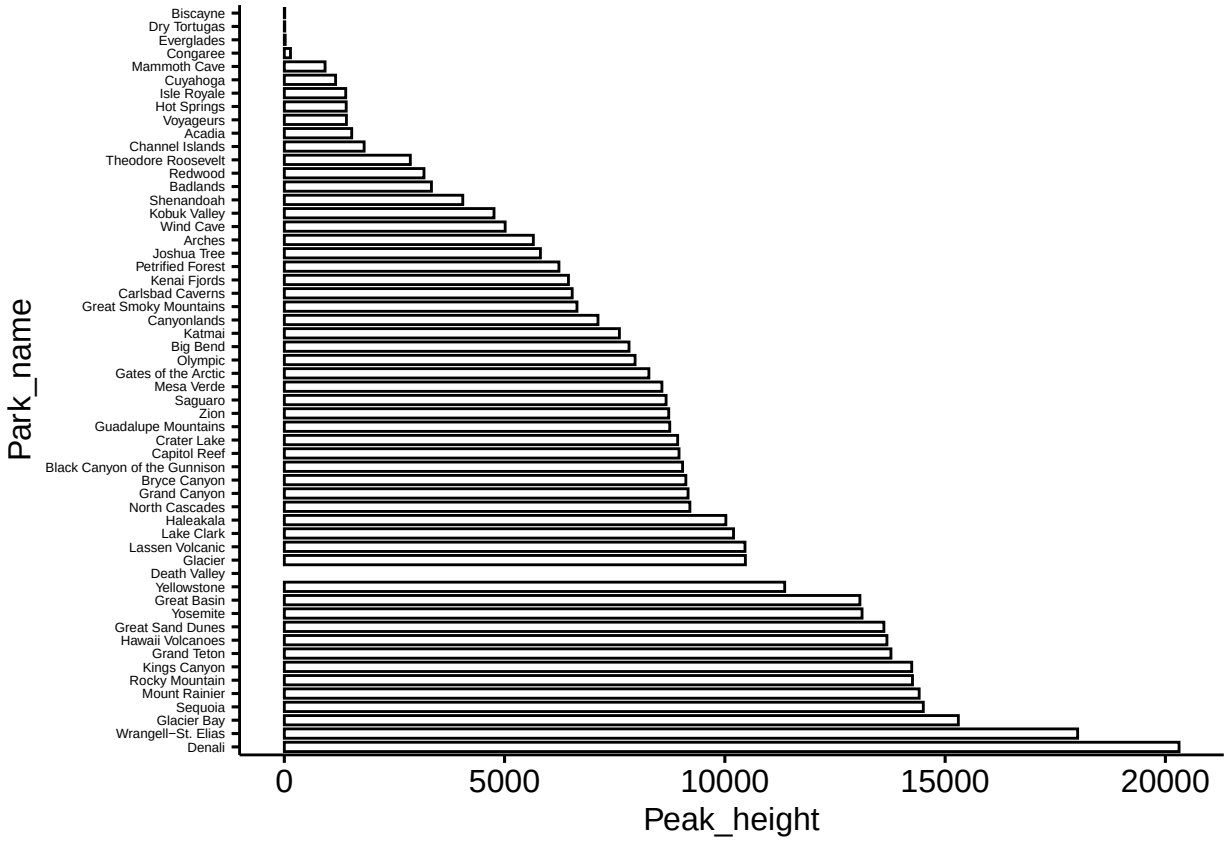
```
## Warning: NAs introduced by coercion
```

```
#final clean up of the datafram to contain the correct formatting, as well as
#trim away any leading and lagging spaces.
elev_all_parks<- elev_all_parks %>%
  mutate(Park_name = as.character(Park_name)) %>%
  mutate(Peak_name = as.character(Peak_name)) %>%
  mutate(Mountain_range = as.character(Mountain_range)) %>%
  mutate_if(is.character, str_trim)
```

Plot, which demostrates the peak height for the provided national parks

```
#plotting the range of peaks for all the national parks. Changing the size of the
#text for the ticks in order for them to not overlap.
peaks_plot<-ggbarplot(elev_all_parks, x = "Park_name", y = "Peak_height") +
  coord_flip()
peaks_plot + theme(axis.text.y =element_text(size = 5))
```

```
## Warning: Removed 1 rows containing missing values (position_stack).
```



YOSEMITE TRAIL INFORMATION

```
#additional trail information from a different national park - Yosemite.
#The website is read with rvest.
```

```
yose_trails<-read_html("https://www.nps.gov/yose/planyourvisit/valleyhikes.htm")

yose_trail_names<- yose_trails %>%
  html_nodes("#cs_idCell2x1x1 a") %>%
  html_text()

diff_yose_trail<- yose_trails %>%
  html_nodes("tr+ tr td:nth-child(2)") %>%
  html_text()

#similar to the previous
yose_name<-replicate(10, "Yosemite")
yosemite_trails<- data.frame(yose_name, yose_trail_names, diff_yose_trail)
```

Connecting the three dataframes to contain the trail and difficulty information.

```
#cleaning up the final dataframes in order to connect them into one trail information containing datafr
yellow_trail_connect<-yellowstone_trails[-4]
yellow_trail_connect <- as.tibble(yellow_trail_connect)

#making sure that the names are the same as in all other tibbles to connect into one dataframe
yellow_trail_connect <- yellow_trail_connect %>%
  dplyr::rename(park = park_name) %>%
  dplyr::rename(trail_name = yellowstone_trail_name) %>%
  dplyr::rename(difficulty = difficulty_level)



yosemite_trails_connect<- yosemite_trails %>%
  dplyr::rename(park = yose_name) %>%
  dplyr::rename(trail_name = yose_trail_names) %>%
  dplyr::rename(difficulty = diff_yose_trail )
```

```
#connecting the dataframes
trail_level<-rbind(yellow_trail_connect, yosemite_trails_connect)
head(trail_level)
```

```
## # A tibble: 6 x 3
##   park        trail_name                              difficulty
##   <fct>       <chr>                                   <chr>
## 1 Yellowstone Storm point loop trail                  " Easy"
## 2 Yellowstone Trout lake                              " Easy"
## 3 Yellowstone Mystic falls                            " Easy"
## 4 Yellowstone Tower fall overlook                     " Easy"
## 5 Yellowstone Uncle tom's trail (from artist point) " Moderate"
## 6 Yellowstone Elephant back mountain trail            " Moderate"
```

```
#the third data subset to add to the final trail iformation containing dataframe
olympic_trails<-read_csv("nationalparked-olympic-data-(Crawl-Run)---2019-12-01T191803Z.csv")
```

```
## Parsed with column specification:
## cols(
##   url = col_character(),
##   trail_name = col_character(),
##   difficulty = col_character()
## )
```

```
olympic_trails<- olympic_trails[,-c(1)]
olymp_names<-replicate(9, "Olympic")
olympic_trails_connect<-data.frame(olymp_names, olympic_trails)
olympic_trails_connect <- olympic_trails_connect %>%
  dplyr::rename(park = olymp_names)
head(olympic_trails_connect)
```

```
##      park                    trail_name       difficulty
## 1 Olympic   Ancient Groves Nature Trail             Easy
## 2 Olympic                Hall of Mosses             Easy
## 3 Olympic Hurricane Ridge Meadow Trails Easy to Moderate
## 4 Olympic           Madison Creek Falls        Very Easy
## 5 Olympic                Marymere Falls  Moderately Easy
## 6 Olympic                  Rialto Beach         Moderate
```

```
#adding the third data subset to the previosuly made up dataframe with trail information
trail_level_parks<- rbind(trail_level, olympic_trails_connect)
#figuring out which difficulty levels need to be fomratted
trail_level_parks %>%
  group_by(difficulty) %>%
  count()
```

```
## # A tibble: 10 x 2
## # Groups:   difficulty [10]
##    difficulty                n
##    <chr>                 <int>
##  1 " Easy"                   5
##  2 " Moderate"               4
##  3 " Strenuous"              3
##  4 Easy                      6
##  5 Easy to Moderate          2
##  6 Moderate                  3
##  7 Moderate to Strenuous     2
##  8 Moderately Easy           2
##  9 Strenuous                 3
## 10 Very Easy                 1
```

```
#formatting the difficulty level to contain universal names
trail_level_3<-trail_level_parks %>%
  mutate(difficulty = str_replace_all(difficulty, " ", "")) %>%
  mutate(difficulty = str_replace_all(difficulty, "ModeratelyEasy", "EasytoModerate")) %>%
  mutate(difficulty = str_replace_all(difficulty, "VeryEasy", "Easy")) %>%
  mutate(difficulty = str_replace_all(difficulty, "EasytoModerate", "Moderate")) %>%
  mutate(difficulty = str_replace_all(difficulty, "ModeratetoStrenuous", "Strenuous"))
#doubel checking that there are only desired level of difficulty.
trail_level_3 %>%
  group_by(difficulty) %>%
  count()
```

```
## # A tibble: 3 x 2
## # Groups:   difficulty [3]
##   difficulty     n
##   <chr>      <int>
## 1 Easy          12
## 2 Moderate      11
```

```
## 3 Strenuous      8
```

```
trail_level_3<- trail_level_3 %>%
  mutate_if(is.character, str_trim)
head(trail_level_3)
```

```
## # A tibble: 6 x 3
##   park        trail_name                              difficulty
##   <fct>       <chr>                                   <chr>
## 1 Yellowstone Storm point loop trail                  Easy
## 2 Yellowstone Trout lake                              Easy
## 3 Yellowstone Mystic falls                            Easy
## 4 Yellowstone Tower fall overlook                     Easy
## 5 Yellowstone Uncle tom's trail (from artist point) Moderate
## 6 Yellowstone Elephant back mountain trail          Moderate
```

Oldest parks:

can compare with the previous publicly available data.

```
#Reading in the web page with the 10 oldest parks in order to compare
#it to the wikipedia provided information
oldest<-read_html("https://www.worldatlas.com/articles/oldest-national-parks-in-the-united-states.html")

#extracting the nodes with the necessary information.
oldest_names<- oldest %>%
  html_nodes("td:nth-child(2)") %>%
  html_text()

date_oldest<- oldest %>%
  html_nodes("td~ td+ td") %>%
  html_text()
#uniting as a dataframe.
old_parks<- data.frame(oldest_names, date_oldest)
#cleaning up the dataframe to separate the park and the state information into different columns.
old_parks<-old_parks %>%
  separate(., oldest_names, into = c("Park","State"), sep = ",")
#cleaning up the dates column in order to be able to format it into the date format.
old_park_2<- old_parks %>%
  mutate(State = str_replace(State,"Wyoming-Montana-", "")) %>%
  mutate(date_oldest = str_replace_all(date_oldest, "st|th|nd", "")) %>%
  mutate(date_oldest = mdy(date_oldest))

#performing the same format change in order to upload it into the SQL database.
old_park_2$date_oldest<- as.character(as.Date(as.character(old_park_2$date_oldest), format = "%Y-%m-%d")
#trimming out the leading and lagging spaces.
old_park<-old_park_2 %>%
  mutate_if(is.character, str_trim)
head(old_park_2)
```

```
##            Park         State date_oldest
## 1   Yellowstone         Idaho  1872-03-01
## 2       Sequoia    California  1890-09-25
## 3      Yosemite    California  1890-10-01
## 4 Mount Rainier    Washington  1899-03-02
```

```
## 5   Crater Lake        Oregon  1902-05-22
## 6     Wind Cave  South Dakota  1903-01-09
```

RECREATIONAL VISITS FOR 10 YEARS

```
#reading in the obtained annual visitations report.
annual_report<-read_excel("Annual_park_1979_now.xlsx", col_names = TRUE, skip = 6 )
```

```
## New names:
## * `` -> ...1
## * `` -> ...2
## * `` -> ...3
## * `` -> ...4
## * `` -> ...5
## * ... and 9 more problems
```

```
#cleaning up the unnecesasry artificats of reading in the excel file into R
annual_report<-annual_report[-1, -c(11,12)]
#performing this name change for now to keep easy track of changed instead of numbers.
colnames(annual_report)<- c("Park_name", "one", "two", "three", "four", "five", "six", "sev", "eight",
```

```
#changing the layout and the structure of the dataframe to assign different year
#values to the rows instead of the seprate columns
annual_report<- gather(annual_report, Year, Visitors, one:tin)
#performign the name change back tot he years
annual_report <- annual_report %>%
  mutate(Year = str_replace(Year, "one", "2009")) %>%
  mutate(Year = str_replace(Year, "two", "2010"))  %>%
  mutate(Year = str_replace(Year, "three", "2011"))  %>%
  mutate(Year = str_replace(Year, "four", "2012"))  %>%
  mutate(Year = str_replace(Year, "five", "2013"))  %>%
  mutate(Year = str_replace(Year, "six", "2014"))  %>%
  mutate(Year = str_replace(Year, "sev", "2015"))  %>%
  mutate(Year = str_replace(Year, "eight", "2016"))  %>%
  mutate(Year = str_replace(Year, "nine", "2017"))  %>%
  mutate(Year = str_replace(Year, "tin", "2018"))

#formatting the values in the Average column to read as integer and round it as
#some of the values have too many significant values
annual_report<- annual_report %>%
  mutate(Average = as.integer(Average)) %>%
  mutate(Average = format(round(Average, 2), nsmall = 2))
```

```
#formatting the types of columns
annual_report$Year <- as.numeric(annual_report$Year)
annual_report$Average <- as.numeric(annual_report$Average)
head(annual_report)
```

```
## # A tibble: 6 x 4
##   Park_name                      Average  Year Visitors
##   <chr>                            <dbl> <dbl>    <dbl>
## 1 Abraham Lincoln Birthplace NHP  206815  2009   221111
## 2 Acadia NP                      2751707  2009  2227698
## 3 Adams NHP                       210647  2009   253656
## 4 African Burial Ground NM         71389  2009       NA
```

```
## 5 Agate Fossil Beds NM             14868  2009    12694
## 6 Alibates Flint Quarries NM        5160  2009     2918
```

```
#modifying the internal formatting of the names of the parks to make sure
#that the names are recognizable in different table in the SQL database.
annual_report <- annual_report %>%
  mutate(Park_name = str_replace_all(Park_name, "NHP|NM|NP|NHS|NRA|EM|NS|MEM|RES|NNRA|NSR|NS|NRES|NB|& P
  mutate(Park_name = str_replace_all(Park_name, "Shiloh P", "Shiloh")) %>%
  mutate(Park_name = str_replace_all(Park_name, "Pea Ridge P", "Pea Ridge")) %>%
  mutate(Park_name = str_replace_all(Park_name, "Ozark R", "Ozark")) %>%
  mutate(Park_name = str_replace_all(Park_name, "Niobrara R", "Niobrara")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Chickamauga & Chattanooga P", "Chickamauga & Chattanooga
  mutate(Park_name = str_replace_all(Park_name, "Fredericksburg & Spotsylvania P", "Fredericksburg & Spo
  mutate(Park_name = str_replace_all(Park_name, "Gettysburg P", "Gettysburg")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Guilford Courthouse P", "Guilford Courthouse"))%>%
  mutate(Park_name = str_replace_all(Park_name,"Horseshoe Bend P", "Horseshoe Bend")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Kennesaw Mountain P" ,"Kennesaw Mountain")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Kings Mountain P", "Kings Mountain")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Manassas P",  "Manassas"))%>%
  mutate(Park_name = str_replace_all(Park_name,"Richmond P", "Richmond" ))%>%
  mutate(Park_name = str_replace_all(Park_name,"River Raisin P", "River Raisin")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Saint Croix R" , "Saint Croix"))%>%
  mutate(Park_name = str_replace_all(Park_name,"S&RR" , "")) %>%
  mutate(Park_name = str_replace_all(Park_name,"Vicksburg P", "Vicksburg"))
```

```
#trim out all the unnecessary lagging and leading spaces that prevent
#tables inteh database from joining on the same values of the park name.
annual_report<-annual_report %>%
  mutate_if(is.character, str_trim)
head(annual_report)
```

```
## # A tibble: 6 x 4
##   Park_name                   Average  Year Visitors
##   <chr>                         <dbl> <dbl>    <dbl>
## 1 Abraham Lincoln Birthplace  206815  2009   221111
## 2 Acadia                     2751707  2009  2227698
## 3 Adams                       210647  2009   253656
## 4 African Burial Ground        71389  2009       NA
## 5 Agate Fossil Beds            14868  2009    12694
## 6 Alibates Flint Quarries       5160  2009     2918
```

Example of the kind of visualisation that can be used to address the chnages in the visitor number for different parks

```
#plot example of the analysis that can be done on visito number to certain parks for all the years.
zio_plot<-annual_report %>%
  filter(Park_name == "Zion") %>%
  ggplot() + geom_line(aes(x = Year, y = Visitors )) +
  ggtitle("Zion visitor number over the years")
```

```
#subsetting the data arbitrary to only contain the to 100 most visited
#parks irrespective of the park and year limitations - done for exampel visualizaztion purposes.
plot100_most<-annual_report %>%
  group_by(Year) %>%
  arrange(desc(Visitors)) %>%
  head(100)
```

Demonstration of the most visited parks over the years, limited sample of data for this example plot.

```
#another plot as an example of the type of visualization that can be done on the changes in
#visitors to the parks - in this case subsetting the data arbitrary at 100 most visited parks
#all through out the ten year reported period, Becasue of this arbitrary cut off, some parks
#only have sporadic values, as they reached the significant level
top100_plot<-ggplot(plot100_most, aes(x = Year, y = Visitors, color = Park_name)) +
  geom_line() +
  ggtitle("Visitor number over the years in top visited\n National Parks in the US ")
```

Initate the database creation

```
db <- dbConnect(SQLite(), dbname="NationalParkInfo.sqlite")
dbSendQuery(conn = db, "pragma foreign_keys=on;")
```

```
## <SQLiteResult>
##   SQL  pragma foreign_keys=on;
##   ROWS Fetched: 0 [complete]
##        Changed: 0
```

Clean parks

```
#creating tables, identifying the primary key
# dbSendQuery(conn = db,"CREATE TABLE clean_parks (
#                         Name TEXT,
#                         State TEXT,
#                         Date_established DATE,
#                         Area_acres INTEGER,
#                         PRIMARY KEY (Name))
#                         WITHOUT ROWID")
```

```
#inserting the data from the appropriate tables in to the
# dbWriteTable(conn = db, name = "clean_parks", value = clean_parks, row.names=FALSE, append = TRUE)
```

```
#example of the equery working on the provided data -
clean_park_q<-dbGetQuery(db, "SELECT * FROM clean_parks")
```

```
## Warning: Closing open result set, pending rows
```

```
## Warning in result_fetch(res@ptr, n = n): Column `Area_acres`: mixed type, first
## seen values of type integer, coercing other values of type string
```

```
head(clean_park_q)
```

```
##             Name          State Date_established Area_acres
## 1         Acadia          Maine       1919-02-26    4907526
## 2 American Samoa American Samoa       1988-10-31     825667
## 3         Arches           Utah       1971-11-12    7667898
## 4       Badlands   South Dakota       1978-11-10   24275594
## 5       Big Bend          Texas       1944-06-12   80116321
## 6       Biscayne        Florida       1980-06-28   17297111
```

Annual Report

```
# dbSendQuery(conn = db,"CREATE TABLE annual_report (
#                         Park_name TEXT,
#                         Average Integer,
#                         Year INTEGER,
#                         Visitors INTEGER,
```

```
#                         PRIMARY KEY (Park_name, Year))
#                         WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "annual_report", value = annual_report, row.names=FALSE, append = TRUE
```

Visist

```
# dbSendQuery(conn = db,"CREATE TABLE most_visits  (
#                         Park_name TEXT,
#                         visitor_number INTEGER,
#                         PRIMARY KEY (Park_name))
#                         WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "most_visits", value = most_visits, row.names=FALSE, append = TRUE)
```

Elevation

```
  # dbSendQuery(conn = db,"CREATE TABLE elev_all_parks (
  #                         Park_name TEXT,
  #                         Peak_name TEXT,
  #                         Mountain_range TEXT,
  #                         Min_elevation_feet INTEGER,
  #                         Vertical_relief_feet INTEGER,
  #                         Peak_height INTEGER,
  #                         PRIMARY KEY (Park_name))
  #                         WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "elev_all_parks", value = elev_all_parks, row.names=FALSE, append = TR
```

Trail difficulty

```
# dbSendQuery(conn = db,"CREATE TABLE trail_level_3 (
#                         park TEXT,
#                         trail_name TEXT,
#                         difficulty TEXT,
#                         PRIMARY KEY (trail_name))
#                         WITHOUT ROWID")
```

```
# dbWriteTable(conn = db, name = "trail_level_3", value = trail_level_3, row.names=FALSE, append = TRUE
```

```
# dbSendQuery(conn = db,"CREATE TABLE old_park_2 (
#                         Park TEXT,
#                         State TEXT,
#                         date_oldest DATE,
#                         PRIMARY KEY (Park))
#                         WITHOUT ROWID")
```

```
#dbWriteTable(conn = db, name = "old_park_2", value = old_park_2, row.names=FALSE, append = TRUE)
```

```
#visualization of the tables in the database.
dbListTables(db)
```

```
## [1] "annual_report"  "clean_parks"    "elev_all_parks" "most_visits"
## [5] "old_park_2"     "trail_level_3"
```

```
#This gets the visitors number for 2009 from the National Park Services
#for only national parks in the US reported by the Wikipedia
all_visits_2009<-dbGetQuery(db, "SELECT annual_report.Park_name, annual_report.Visitors, annual_report.
          FROM annual_report
```

```
          INNER JOIN clean_parks on clean_parks.Name = annual_report.Park_name
          WHERE annual_report.Year = 2009")
head(all_visits_2009)
```

```
##                       Park_name Visitors Year
## 1                        Acadia  2227698 2009
## 2                        Arches   996312 2009
## 3                      Badlands   933918 2009
## 4                      Big Bend   363905 2009
## 5                      Biscayne   437745 2009
## 6 Black Canyon of the Gunnison   171451 2009
```

```
#this shows that the names are the same in the annual_report and clean_parks,
#and that SQL queries will work on these tables when they need to be joined.
same_name<-dbGetQuery(db, "SELECT annual_report.Park_name FROM annual_report INTERSECT SELECT clean_par
head(same_name)
```

```
##                       Park_name
## 1                        Acadia
## 2                        Arches
## 3                      Badlands
## 4                      Big Bend
## 5                      Biscayne
## 6 Black Canyon of the Gunnison
```

```
#This demonstrates that the names are identical in other dataframes as well.
old_park_query<-dbGetQuery(db, "SELECT old_park_2.park FROM old_park_2 INTERSECT SELECT annual_report.Pa
head(old_park_query)
```

```
##             Park
## 1    Crater Lake
## 2        Glacier
## 3      Haleakala
## 4     Mesa Verde
## 5  Mount Rainier
## 6 Rocky Mountain
```

```
library(maps)
```

```
##
## Attaching package: 'maps'
```

```
## The following object is masked from 'package:purrr':
##
##     map
```

```
usa <- map_data("usa")
```

```
#creating a separate column that has the abbreaviation of the states in order to be able to plot it wit
clean_parks_abb<- clean_parks
clean_parks_abb$State_abb<-state.abb[match(clean_parks_abb$State,state.name)]

plot_us_clean<-clean_parks_abb %>%
  dplyr::rename(State_name = State)  %>%
  dplyr::rename(state = State_abb)
```

Connect with annual report with clean data info and plot the visitor population.

```
#I used this quaery as an example for future work that this database can be used for -
#specifically, verification of the data obtained from open web soruces. T
#his cross reference verification is can be used with dates and other data.
compare_dates<-dbGetQuery(db, "SELECT clean_parks.Name, clean_parks.State, clean_parks.Date_established
           FROM clean_parks
            INNER JOIN old_park_2 on old_park_2.Park = clean_parks.Name ")
head(compare_dates)
```

```
##             Name      State Date_established date_oldest          Park
## 1    Crater Lake     Oregon       1902-05-22  1902-05-22    Crater Lake
## 2        Glacier    Montana       1910-05-11  1910-05-11        Glacier
## 3     Mesa Verde   Colorado       1906-06-29  1906-06-29     Mesa Verde
## 4  Mount Rainier Washington       1899-03-02  1899-03-02  Mount Rainier
## 5 Rocky Mountain   Colorado       1915-01-26  1915-01-26 Rocky Mountain
## 6        Sequoia California       1890-09-25  1890-09-25        Sequoia
```

```
#combining the created two dataframes by the names of the park in order
#to plot the average number of visitors
full_state_visit<-inner_join(clean_parks_abb, annual_report, by = c("Name" = "Park_name"))
```

```
#This plot visualizes the average number of visitors per year to the national parks
# in the corresponding states over the period of 10 years.
full_state_visit<-full_state_visit %>%
  dplyr::rename(state = State_abb)
plot_usmap(data = full_state_visit, values = "Average", labels = TRUE)
```